add.phenos

Add phenotypes to cross object.

Description

Add phenotypes to cross object by checking index.

Usage

add.phenos(cross, newdata = NULL, index = NULL)

Arguments

cross object of class cross; see read.cross
newdata data frame with row names matching values of phenotype identified by index for object cross
index character string name of phenotype in object cross; if NULL, then newdata must be of same size as cross with phenotypes in order

Details

The name index must be a phenotype in the cross object. The row names of newdata are matched with values of index.

Value

object of class cross with added phenotypes

Author(s)

Brian S. Yandell, <byandell@wisc.edu>

See Also

read.cross

Examples

## Not run:
data(hyper)
x <- data.frame(x = rnorm(nind(hyper)))
hyperx <- add.phenos(hyper, x)

## End(Not run)
**CMSTTests**

**CMSTTests**  
*Perform CMST Tests on cross object*

### Description

Performs 6 separate CMST tests (3 versions, 2 penalties).

### Usage

```r
CMSTtests(cross, phenol, pheno2, Q.chr, Q.pos,  
  addcov1 = NULL, addcov2 = NULL, intcov1 = NULL, intcov2 = NULL,  
  method = c("par", "non.par", "joint", "all"),  
  penalty = c("bic", "aic", "both"), verbose = FALSE)
```

### Arguments

- `cross`: object of class `cross`
- `phenol`: first phenotype column number or character string name
- `pheno2`: second phenotype column number or character string name; if more than one, then all phenotypes will be tested against `phenol`
- `Q.chr`: QTL chromosome (number or label)
- `Q.pos`: QTL position in cM
- `addcov1`, `addcov2`: additive covariates for first and second phenotype, respectively
- `intcov1`, `intcov2`: interactive covariates for first and second phenotype, respectively
- `method`: test method; see details
- `penalty`: type of penalty; see details
- `verbose`: verbose printout if TRUE

### Details

Explain method and penalty here.

### References


### See Also

`CMSTCross`, `PrecTpFpMatrix`, `FitAllTests`
Examples

```r
data(CMSTCross)
nms <- names(CMSTCross$pheno)
out1 <- CMSTests(CMSTCross,
    pheno1 = nms[1],
    pheno2 = nms[2],
    Q.chr = 1,
    Q.pos = 55,
    addcov1 = NULL,
    addcov2 = NULL,
    intcov1 = NULL,
    intcov2 = NULL,
    method = "all",
    penalty = "both")

out1[1:6]
out1[7]
out1[8:12]
out1[13:17]
```

## list of phenotypes
```r
out2 <- CMSTests(CMSTCross,
    pheno1 = nms[1],
    pheno2 = nms[-1],
    Q.chr = 1,
    Q.pos = 55,
    addcov1 = NULL,
    addcov2 = NULL,
    intcov1 = NULL,
    intcov2 = NULL,
    method = "par",
    penalty = "bic")
```

```
filter.threshold  
Summary of threshold results

Description

Summary of threshold results.

Usage

`filter.threshold(cross, pheno.col, latent.eff, res.var, lod.thrs, drop.lod = 1.5, s.quant, n.perm, alpha.levels, qh.thrs, ww.thrs, addcovar = NULL, intcovar = NULL, verbose = FALSE, ...)`

Arguments

cross  object of class cross; see `read.cross`
pheno.col phenotype columns used for filtering thresholds
filter.threshold

latent.eff  ratio of latent effect SD to residual SD
res.var     residual variance (=SD^2)
lod.thrs    LOD threshold values for range of significance (alpha) levels
drop.lod    LOD drop from max LOD to keep in analysis
s.quant    vector of 1:Nmax with Nmax the maximum hotspot size to be considered
n.perm      number of permutations
alpha.levels range of significance levels; same length as lod.thrs
qh.thrs     Results of call to hotperm
ww.thrs     Results of call to ww.perm
addcovar    additive covariates as vector or matrix; see scanone
intcovar    interactive covariates as vector or matrix; see scanone
verbose     verbose output if TRUE
...         arguments passed along to scanone

Value

List with items

NL.thrs
N.thrs
WW.thrs
NL
N.counts
WW.counts

References


See Also

hotperm, ww.perm, scanone
GetCandReg

Get genetic information on candidate regulators and co-mapping traits.

Description
Get chromosome (phys.chr) and physical position in cM (phys.pos), along with the LOD score (peak.lod) at the peak position (peak.pos), and the chromosome where the peak is located (peak.chr). Some candidates may map to the same chromosome where they are physically located.

Usage
GetCandReg(highobj, annot, traits)
GetCisCandReg(highobj, cand.reg, lod.thr = NULL)
GetCoMappingTraits(highobj, cand.reg)

Arguments
highobj data frame from highlod, which is sparse summary of high LODs in large scanone object
annot data frame with annotation information; must have first column as unique identifier, third column as chromosome, and fifth column as position in cM; typically column 2 has gene name, and column 4 has position in Mb
traits names of traits to examine as candidate regulators; names must correspond to phenotypes in cross object
cand.reg data frame with candidate regulator; see value section below
lod.thr LOD threshold; restrict to intervals above this value if not NULL

Details
Traits that map to positions close to their physical locations are said to map in cis (local linkages). Traits that map to positions away from their physical locations are said to map in trans (distal linkages). There is no unambiguous way to determine how close a trait needs to map to its physical location in order to be classified as cis. Our choice is to classify a trait as cis if the 1.5-LOD support interval (Manichaikul et al. 2006) around the LOD peak contains the trait’s physical location, and if the LOD score at its physical location is higher the the LOD threshold. The function GetCisCandReg determines which of the candidate regulators map in cis. The function GetCoMappingTraits returns a list with the putative targets of each trait. A trait is included in the putative target list of a trait when its LOD peak is greater than lod.thr and the drop LOD support interval around the peak contains the location of the trait’s QTL. The function jointestoutputs currently relies on external files that contain results of FitAllTests. It needs to be rewritten to save space.
**Value**

GetCoMappingTraits returns a list with each element being the names of co-mapping traits for a particular name in traits. GetCandReg returns a data frame while GetCisCandReg returns a list with a similar candidate regulator data frame as the element cis.reg, and the index of trait names as the element cis.index. The elements of the candidate regulator data frame are as follows (peak.pos.lower and peak.pos.upper only for GetCisCandReg):

- **gene** name of trait, which might be a gene name
- **phys.chr** chromosome on which gene physically resides
- **phys.pos** physical position (in cM)
- **peak.chr** chromosome where peak LOD is located
- **peak.pos** position of peak (in cM)
- **peak.lod** LOD value at peak
- **peak.pos.lower, peak.pos.upper** lower and upper bounds of the 1.5-LOD support interval around peak.pos

**Author(s)**

Elias Chaibub Neto

**References**

Manichaikul et al. (2006) Genetics

**See Also**

highlod, FitAllTests, scanone

**Examples**

```r
# data(CMSTCross) is loaded lazily.
CMSTscan <- scanone(CMSTCross, pheno.col = 1:3, method = "hk")
CMSThigh <- highlod(CMSTscan)
traits <- names(CMSTCross$pheno)
annot <- data.frame(name = traits, traits = traits, chr = rep(1, 3),
Mb.pos = c(55, 10, 100))
annot$H.pos <- annot$Mb.pos

cand.reg <- GetCandReg(CMSThigh, annot, traits)
cis.cand.reg <- GetCisCandReg(CMSThigh, cand.reg)
comap.targets <- GetCoMappingTraits(CMSThigh, cand.reg)
```
GetCommonQtls

Get common QTLs for phenotypes

Description

Perform joint QTL mapping for phenotypes with marginal LOD peak positions higher than LOD threshold and within set distance of each other.

Usage

GetCommonQtls(cross, pheno1, pheno2, thr = 3, peak.dist = 5,
               addcov1 = NULL, addcov2 = NULL, intcov1 = NULL, intcov2 = NULL)

Arguments

- cross: object of class cross
- pheno1: first phenotype column number or character string name
- pheno2: second phenotype column number or character string name; if more than one, then all phenotypes will be tested against pheno1
- thr: LOD threshold
- peak.dist: maximal peak distance to be considered the same peak (in cM)
- addcov1, addcov2: additive covariates for first and second phenotype, respectively
- intcov1, intcov2: interactive covariates for first and second phenotype, respectively

References


See Also

CMSTCross

Examples

data(CMSTCross)
commqtls <- GetCommonQtls(CMSTCross,
                           pheno1 = "y1",
                           pheno2 = "y3",
                           thr = 3,
                           peak.dist = 5,
                           addcov1 = NULL,
                           addcov2 = NULL,
                           intcov1 = NULL,
                           intcov2 = NULL)

commqtls
highlod

Pull high LOD values with chr and pos.

Description

Pull high LOD values with chr and pos.

Usage

```r
highlod(scans, lod.thr = 0, drop.lod = 1.5,
        extend = TRUE, restrict.lod = FALSE, ...)
pull.highlod(object, chr, pos, ...)
## S3 method for class 'highlod'
print(x, ...)
## S3 method for class 'highlod'
summary(object, ...)
## S3 method for class 'highlod'
plot(x, ..., quant.level = NULL, sliding = FALSE)
## S3 method for class 'highlod'
max(x, lod.thr = NULL, window = NULL, quant.level = NULL, ...)
## S3 method for class 'highlod'
quantile(x, probs = NULL, lod.thr = NULL, n.quant,
          n.pheno, max.quantile = TRUE, ...)
```

Arguments

- `scans`: object of class `scanone`
- `lod.thr`: LOD threshold
- `drop.lod`: LOD drop from max to keep for support intervals
- `extend`: extend support interval just past `drop.lod`; matches behavior of `lodint` when `TRUE`
- `restrict.lod`: restrict to loci above LOD threshold if `TRUE`; matches behavior of `lodint` when `FALSE` (default)
- `chr`: chromosome identifier
- `pos`: position, or range of positions, in cM
- `x,object`: object of class `highlod`
- `probs`: probability levels for quantiles (should be > 0.5)
- `n.quant`: maximum of `s.quant`
- `n.pheno`: number of phenotypes considered
- `max.quantile`: return only quantiles of max LOD across genome if `TRUE`
- `window`: size of window for smoothing hotspot size
- `quant.level`: vector of LOD levels for 1 up to `length(quant.level)` size hotspots
- `sliding`: plot as sliding hotspot if `TRUE`
- `...`: arguments passed along
Details
The highlod condenses a scanone object to the peaks above a lod.thr and/or within drop.lod of such peaks. The pull.highlod pulls out the entries at a particular genomic location or interval of locations. Summary, print, plot, max and quantile methods provide ways to examine a highlod object.

Value
Data frame with
- row: row number in scanone object
- phenos: phenotype column number
- lod: LOD score for phenotype at locus indicated by row

Author(s)
Brian S Yandell and Elias Chaibub Neto

See Also
highlod, hotperm

Examples
```r
example(include.hotspots)
scan1 <- scanone(cross1, pheno.col = 1:1000, method = "hk")
high1 <- highlod(scan1, lod.thr = 2.11, drop.lod = 1.5)
pull.highlod(high1, chr = 2, pos = 24)
summary(high1, lod.thr = 2.44)
max(high1, lod.thr = seq(2.11, 3.11, by = .1))
```

---

hotperm Conduct NL and N permutation tests

Description
Conduct NL and N permutation tests.

Usage
```r
hotperm(cross, n.quant, n.perm, lod.thrs, alpha.levels, drop.lod = 1.5,
        window = NULL, verbose = FALSE, init.seed = 0,
        addcovar = NULL, intcovar = NULL, ...)
data(hotperm1)
## S3 method for class 'hotperm'
print(x, ...)
## S3 method for class 'hotperm'
```
summary(object, quant.levels, ...)  
## S3 method for class 'hotperm'  
quantile(x, probs, ..., lod.thr = NULL)  
## S3 method for class 'summary.hotperm'  
print(x, ...)  

**Arguments**  
cross object of class cross  
n.quant maximum of s.quant  
n.perm number of permutations  
lod.thrs vector of LOD thresholds  
alpha.levels vector of significance levels  
quant.levels quantile levels, as number of traits, to show in summary; default is 1, 2, 5, 10, ... up to maximum recorded  
drop.lod LOD drop amount for support intervals  
window window size for smoothed hotspot size  
verbose verbose output if TRUE  
init.seed initial seed for pseudo-random number generation  
x,object object of class hotperm or summary.hotperm  
probs probability levels for quantiles (1-probs if all > 0.5); default is alpha.levels  
lod.thr restrict to values above this if not NULL  
addcovar additive covariates as vector or matrix; see scanone  
intcovar interactive covariates as vector or matrix; see scanone  
... arguments passed along to scanone  

**Author(s)**  
Elias Chaibub Neto and Brian S Yandell  

**Examples**  
example(include.hotspots)  
set.seed(123)  
pt <- scanone(ncross1, method = "hk", n.perm = 1000)  
alphas <- seq(0.01, 0.10, by=0.01)  
lod.thrs <- summary(pt, alphas)  
## Not run:  
## This takes awhile, so we save the object.  
set.seed(12345)  
hotperm1 <- hotperm(cross = cross1,  
n.quant = 300,  
n.perm = 100,  
lod.thrs = lod.thrs,  
alpha.levels = alphas,  
drop.lod = 1.5,
Description

Determine hotspot sizes and display. Use individual threshold and quantile thresholds as provided.

Usage

```r
hotsize(hotobject, ...)  # S3 method for class 'scanone'
hotsize(hotobject, lod.thr = NULL, drop.lod = 1.5, ...)  # S3 method for class 'highlod'
hotsize(hotobject, lod.thr = NULL, window = NULL, quant.level = NULL, ...)  # S3 method for class 'hotsize'
print(x, ...)  # S3 method for class 'hotsize'
summary(object, ...)  # S3 method for class 'hotsize'
plot(x, ylab = "counts", quant.axis = pretty(x$max.N), col = c("black", "red", "blue"), by.chr = FALSE, maps = NULL, title = "", ...)  # S3 method for class 'hotsize'
```

Arguments

- `hotobject`: object of class `scanone` or `highlod`
- `lod.thr`: LOD threshold
- `drop.lod`: LOD drop from max to keep for support intervals
- `window`: window width in cM for smoothing hotspot size; not used if 0 or NULL
- `quant.level`: vector of LOD levels for 1 up to `length(quant.level)` size hotspots
- `x,object`: object of class `hotsize`
- `ylab`: label for vertical plot axis
- `quant.axis`: hotspot sizes for quantile axis (vertical on right side of plot)
- `col`: color of hotspot size, smoothed hotspot size, and sliding hotspot size
- `by.chr`: separate plot by chromosome if TRUE
- `maps`: if not NULL, list of objects of class `map` to use for rugs on top and bottom of plot
- `title`: title for plot
- `...`: arguments passed along to `scanone` methods
Value

hotsize methods return an object of class hotsize, which is essentially an object of class summary.scanone with additional attributes for lod.thr, window, and quant.level.

Author(s)

Brian S Yandell and Elias Chaibub Neto

See Also

highlod, hotperm

Examples

example(highlod)
hot1 <- hotsize(high1)
summary(hots1)
plot(hots1)

parallel.qtlhot Code for parallelizing R/qtlhot.

Description

Code for parallelizing R/qtlhot. See installed parallel directory for proper use. There is apparently an S3 parallel method, so doc has to be as shown below, even though it is called as parallel.qtlhot.

Usage

## S3 method for class 'qtlhot'
parallel(x, data = 1, ..., dirpath = ".")
qtlhot.phase0(dirpath, init.seed = 92387475, len = rep(400, 16), n.mar = 185, n.ind = 112,
n.phe = 100, latent.eff = 0, res.var = 1, lod.thrs, ...)
big.phase0(dirpath, cross, trait.file, trait.matrix, droptrait.names = NULL,
   keeptrait.names = NULL, lod.thrs, sex = "Sex", trait.index,
   batch.effect = NULL, size.set = 250, offset = 0, subset.sex = NULL, verbose = TRUE)

Arguments

x phase of parallel processing (1,2,3)
data index for parallel processing (1,2,...)
... additional arguments passed along
dirpath directory path as character string
init.seed initial seed for pseudorandom number generation
len vector of chromosome lengths for simulated map
n.mar number of markers for simulated map
PrecTpFpMatrix

- n.ind: number of individuals for simulated cross
- n.phe: number of phenotypes for simulated phenotypes
- latent.eff: size of latent effect
- res.var: magnitude of residual variance
- lod.thrs: vector of LOD thresholds to examine
- cross: object of class cross
- trait.file: character string name of trait file
- trait.matrix: character string name of trait matrix
- droptrait.names: vector of character strings for traits to drop (none if NULL)
- keeptrait.names: vector of character strings for traits to keep (keep all if NULL)
- sex: character string name of phenotype for sex
- trait.index: vector of character strings for trait names
- batch.effect: character string for batch effect (none if NULL)
- size.set: maximum size of set of traits to scan at one time
- offset: offset for name of trait RData files
- subset.sex: string of sex to subset on (both sexes if NULL)
- verbose: verbose output if TRUE

Author(s)

Brian S Yandell and Elias Chaibub Neto

See Also

read.cross

PrecTpFpMatrix

* Determine false positive and true positive rates for known targets.

Description

Determine how well different tests do to predict candidates of regulation.

Usage

FitAllTests(cross, phenol, pheno2, Q.chr, Q.pos, verbose = TRUE)
JoinTestOutputs(comap, tests, file)
PrecTpFpMatrix(alpha, val.targets, all.orfs, tests, cand.reg, cis.cand.reg)
p.adjust.mp(tests, method = "BH")
**Arguments**

- **cross**: object of class `cross`
- **pheno1**: first phenotype column number or character string name
- **pheno2**: second phenotype column number or character string name; if more than one, then all phenotypes will be tested against `pheno1`
- **Q.chr**: QTL chromosome (number or label)
- **Q.pos**: QTL position in cM
- **verbose**: verbose printout if `TRUE`
- **comap**: list result of `GetComappingTraits`
- **alpha**: significance levels at which summaries are computed
- **val.targets**: validated targets of candidate regulators
- **all.orfs**: all trait names
- **tests**: list object as list of `FitAllTests` results, or of joined output created by `JoinTestOutputs`
- **file**: prefix for file names when running `FitAllTests` in parallel and saving test results in separate files
- **cand.reg**: object from `GetCandReg`
- **cis.cand.reg**: object from `GetCisCandReg`
- **method**: method for p-value adjustment; see `p.adjust`

**Details**

`FitAllTests` invokes 7 tests. The hidden routine `CitTests` is invoked by call to `FitAllTests`; this is hidden because we do not recommend its use.

`JoinTestOutputs` joins results of `FitAllTests`, either from a list tests or from a collection of files prefixed by `file`. The joined tests from `JoinTestOutputs` are summarized with `PrecTpFpMatrix` using the biologically validated true positives, false positives and precision, for the inferred causal relations. We define a true positive as a statistically significant causal relation between a gene and a putative target gene when the putative target gene belongs to the known signature of the gene. Similarly, we define a false positive as a statistically significant causal relation between a gene and a putative target gene when the target gene does not belong to the signature. (For the AIC and BIC methods that do not provide a p-value measuring the significance of the causal call, we simply use the detected causal relations in the computation of true and false positives). The validated precision is computed as the ratio of true positives by the sum of true and false positives. The `PrecTpFpMatrix` computes these measures to both all genes, and to cis genes only. Simulations suggest only non-parametric tests need to be adjusted using Benjamini-Hochberg via `p.adjust.pn`.

**Value**

List containing

- **Prec1**, **Prec2**: matrix of precision with rows for significance level and columns for test; first is for all, second is for cis candidates only
- **Tp1**, **Tp2**: matrix of true positive rate with rows for significance level and columns for test; first is for all, second is for cis candidates only
- **Fp1**, **Fp2**: matrix of false positive rate with rows for significance level and columns for test; first is for all, second is for cis candidates only
Author(s)

Elias Chaibub Neto

See Also

GetCandReg, CMSTtests, p.adjust

Examples

```r
example(GetCandReg)
## Suppose y1 is causal with targets y2 and y3.
targets <- list(y1 = c("y2", "y3"))

tests <- list()
for(k in seq(names(comap.targets))) {
  tests[[k]] <- FitAllTests(CMSTCross, pheno1 = names(comap.targets)[k],
                           pheno2 = comap.targets[[k]],
                           Q.chr = cand.reg[k, 4],
                           Q.pos = cand.reg[k, 5])
}

names(tests) <- names(comap.targets)
tests <- JoinTestOutputs(comap.targets, tests)

PrecTpFpMatrix(alpha = seq(0.01, 0.10, by = 0.01),
               val.targets = targets, all.orfs = CMSThigh$names, tests = tests,
               cand.reg = cand.reg, cis.cand.reg = cis.cand.reg)
```

Description

Wrapper routine for simulations.

Usage

```r
sim.hotspot(nSim, cross, n.pheno, latent.eff, res.var = 1, n.quant, n.perm,
            alpha.levels, lod.thrs, drop.lod = 1.5, verbose = FALSE)
mySimulations(...)  
sim.null.cross(chr.len = rep(400, 16), n.mar = 185, n.ind = 112,
               type = "bc", n.pheno = 6000, latent.eff = 1.5, res.var = 1,
               init.seed = 92387475)

sim.null.pheno.data(cross, n.pheno, latent.eff, res.var)

include.hotspots(cross, hchr, hpos, hsize, Q.eff, latent.eff,
                  lod.range.1, lod.range.2, lod.range.3, res.var=1, n.pheno, init.seed)
```
Arguments

- **nSim** Number of simulated sets of phenotypes to create. See details.
- **cross** Object of class `cross`. See `read.cross`.
- **n.pheno** Number of traits, or phenotypes, to simulate for cross object.
- **latent.eff** Strength of latent effect, which is included in all traits. See `sim.null.cross`.
- **res.var** Residual variance for traits. Should not affect results.
- **n.quant** Maximum size of hotspots examined; ideally large enough to exceed the largest Breitling alpha critical value.
- **n.perm** Number of permutations to perform per realization. Good idea to do 1000, but this takes time.
- **alpha.levels** Vector of significance levels.
- **lod.thrs** Vector of LOD thresholds, typically single-trait permutation thresholds for various significance levels.
- **drop.lod** Drop in LOD score examined. LODs below this drop from the maximum for a chromosome will not be scored.
- **init.seed** Initial seed for pseudo-random number generation.
- **chr.len** Vector of chromosome lengths.
- **n.mar** Number of markers.
- **n.ind** Number of individuals.
- **type** Type of cross.
- **hchr, hpos, hsize** Vectors for hotspot chromosomes, positions, and sizes.
- **Q.eff** QTL effect.
- **lod.range.1, lod.range.2, lod.range.3** 2-vectors of LOD ranges for multiple purposes.
- **verbose** Verbose output if TRUE. More detailed output if 2.
- **...** Arguments passed directly to `sim.hotspot`.

Details

Simulate `nSim` realizations of cross object with `n.pheno` phenotypes with correlation `latent.eff`. All simulations use the same genotypes in the `cross` object.

Value

- `sim.null.cross` simulates an object of class `cross`. `sim.null.pheno.data` simulates a data frame of phenotypes. `sim.hotspot` uses these other routines to simulate a hotspot, returning a list object.

Author(s)

Elias Chaibub Neto and Brian S. Yandell
SimCrossCausal

Simulate Cross for Causal Tests

Description

Creates cross with certain pattern of dependence across phenotypes.

Usage

SimCrossCausal(n.ind, len, n.mar, beta, add.eff, dom.eff,
    sig2.1 = 1, sig2.2 = 1, eq.spacing = FALSE,
    cross.type = c("bc", "f2"), normalize = FALSE)
SimCrossIndep(n.ind, len, n.mar, beta, add.eff.1, dom.eff.1,
    add.eff.h, dom.eff.h, sig2.1 = 1, sig2.2 = 1, sig2.h = 1,
    eq.spacing = FALSE, cross.type = "f2", normalize = FALSE)
data(CMSTCross)

Arguments

n.ind number of individuals to simulate
len vector specifying the chromosome lengths (in cM)
n.mar vector specifying the number of markers per chromosome
### References


### Examples

```r
set.seed(987654321)
CMSTCross <- SimCrossCausal(n.ind = 100,
    len = rep(100, 3), n.mar = 101,
    beta = rep(0.5, 2), add.eff = 1, dom.eff = 0,
    sig2.1 = 0.4, sig2.2 = 0.1, eq.spacing = FALSE,
    cross.type = "bc", normalize = TRUE)
CMSTCross <- calc.genoprob(CMSTCross, step = 1)
## Not run:
save(CMSTCross, file = "CMSTCross.RData", compress = TRUE)

## End(Not run)
```

### Description

Conduct West-Wu (Q) permutation tests.

### Usage

```r
ww.perm(highobj, n.perm, lod.thrs, alpha.levels, verbose = FALSE)
```

## S3 method for class 'ww.perm'

```r
print(x, ...)
```

## S3 method for class 'ww.perm'

```r
summary(object, alpha.levels, ...)
```
Arguments

- `highobj`: object of class `highlod`
- `n.perm`: number of permutations
- `lod.thrs`: vector of LOD thresholds
- `alpha.levels`: vector of significance levels
- `x.object`: object of class `ww.perm`
- `...`: ignored
- `verbose`: verbose output if TRUE

Details

Perform permutation tests to assess the statistical significance of the hotspots detected using the West-Wu \( q \)-method permutations. The `ww.perm` function implements the \( q \)-method’s permutation scheme (see the Method’s section of Chaibub Neto et al. 2012, for details). The `n.perm` parameter specifies the number of simulations. Here we set it to 100 in order to save time. In practice, we recommend at least 1,000 permutations. The function’s output is a matrix with 100 rows representing the permutations, and 10 columns representing the QTL mapping thresholds. Each entry \( i,j \), represents the maximum number of significant linkages across the entire genome detected at permutation \( i \), using the LOD threshold \( j \). The `ww.summary` function computes the \( q \)-method’s hotspot size permutation thresholds, that is, the \( 1-\alpha \) quantiles for each one of the QTL mapping LOD thresholds in `lod.thrs`. For instance, the entry at row 10 and column 1 of the `qQNthr` matrix tells us that the 99% percentile of the permutation distribution of genome-wide maximum hotspot size based on a QTL mapping threshold of 2.11 is 27.00. In other words, any hotspot greater than 27 is considered statistically significant at a 0.01 significance level when QTL mapping is done using a 2.11 LOD threshold. In general, we are often interested in using the same error rates for the QTL mapping and hotspot analysis. That is, if we adopt a QTL mapping threshold that controls GWER at a 1% level (in our case, 3.11) we will also want to consider \( \alpha = 0.01 \) for the hotspot analysis, leading to a hotspot threshold of 12.00. Therefore, we are usually more interested in the diagonal of `qQNthr`. We adopted a GWER of 5%, and the corresponding \( q \)-method’s permutation threshold is 18. According to this threshold, all hotspots are significant.

Author(s)

Elias Chaibub Neto and Brian S Yandell

Examples

```r
## Not run:
## All unspecified objects come from vignette qtlhot.
set.seed(12345)
Q.1 <- ww.perm(high1, n.perm = 100, lod.thrs, alphas)
Q.1.thr <- summary(Q.1, alphas)
Q.1.thr
diag(Q.1.thr)

set.seed(12345)
Q.2 <- ww.perm(high2, 100, lod.thrs, alphas)
Q.2.thr <- summary(Q.2, alphas)
```
## End (Not run)
Index

*Topic utilities

add.phenos, 2
CMSTests, 3
filter.threshold, 4
GetCandReg, 6
GetCommonQtls, 8
highlod, 9
hotperm, 10
hotsize, 12
parallel.qtlhot, 13
PrecTpFpMatrix, 14
sim.hotspot, 16
SimCrossCausal, 18
ww.perm, 19

add.phenos, 2

big.phase0 (parallel.qtlhot), 13
CitTests (PrecTpFpMatrix), 14
CMSTCross, 3, 8
CMSTCross (SimCrossCausal), 18
CMSTests, 3, 16
CMSTestsList (CMSTests), 3

filter.threshold, 4
FitAllTests, 3, 6, 7, 15
FitAllTests (PrecTpFpMatrix), 14

GetCandReg, 6, 15, 16
GetCisCandReg, 15
GetCisCandReg (GetCandReg), 6
GetCoMappingTraits (GetCandReg), 6
GetCommonQtls, 8

highlod, 6, 7, 9, 10, 12, 13, 20
hotperm, 5, 10, 10, 13
hotperm1 (hotperm), 10
hotsize, 12

include.hotspots (sim.hotspot), 16

JoinTestOutputs (PrecTpFpMatrix), 14

lodint, 9

max.highlod (highlod), 9
mySimulations (sim.hotspot), 16

p.adjust, 15, 16
p.adjust.np (PrecTpFpMatrix), 14
parallel.qtlhot, 13
plot.highlod (highlod), 9
plot.hotsize (hotsize), 12
PrecTpFpMatrix, 3, 14
print.highlod (highlod), 9
print.hotperm (hotperm), 10
print.hotsize (hotsize), 12
print.summary.hotperm (hotperm), 10
print.ww.perm (ww.perm), 19
pull.highlod (highlod), 9

qtlhot.phase0 (parallel.qtlhot), 13
quantile.highlod (highlod), 9
quantile.hotperm (hotperm), 10

read.cross, 2, 4, 14, 17, 18

scanone, 5–7, 9–12
sim.hotspot, 16
sim.null.cross, 17, 18
sim.null.cross (sim.hotspot), 16
sim.null.pheno.data (sim.hotspot), 16
SimCrossCausal, 18
SimCrossIndep (SimCrossCausal), 18
summary.highlod (highlod), 9
summary.hotperm (hotperm), 10
summary.hotsize (hotsize), 12
summary.scanone, 13
summary.ww.perm (ww.perm), 19

ww.perm, 5, 19